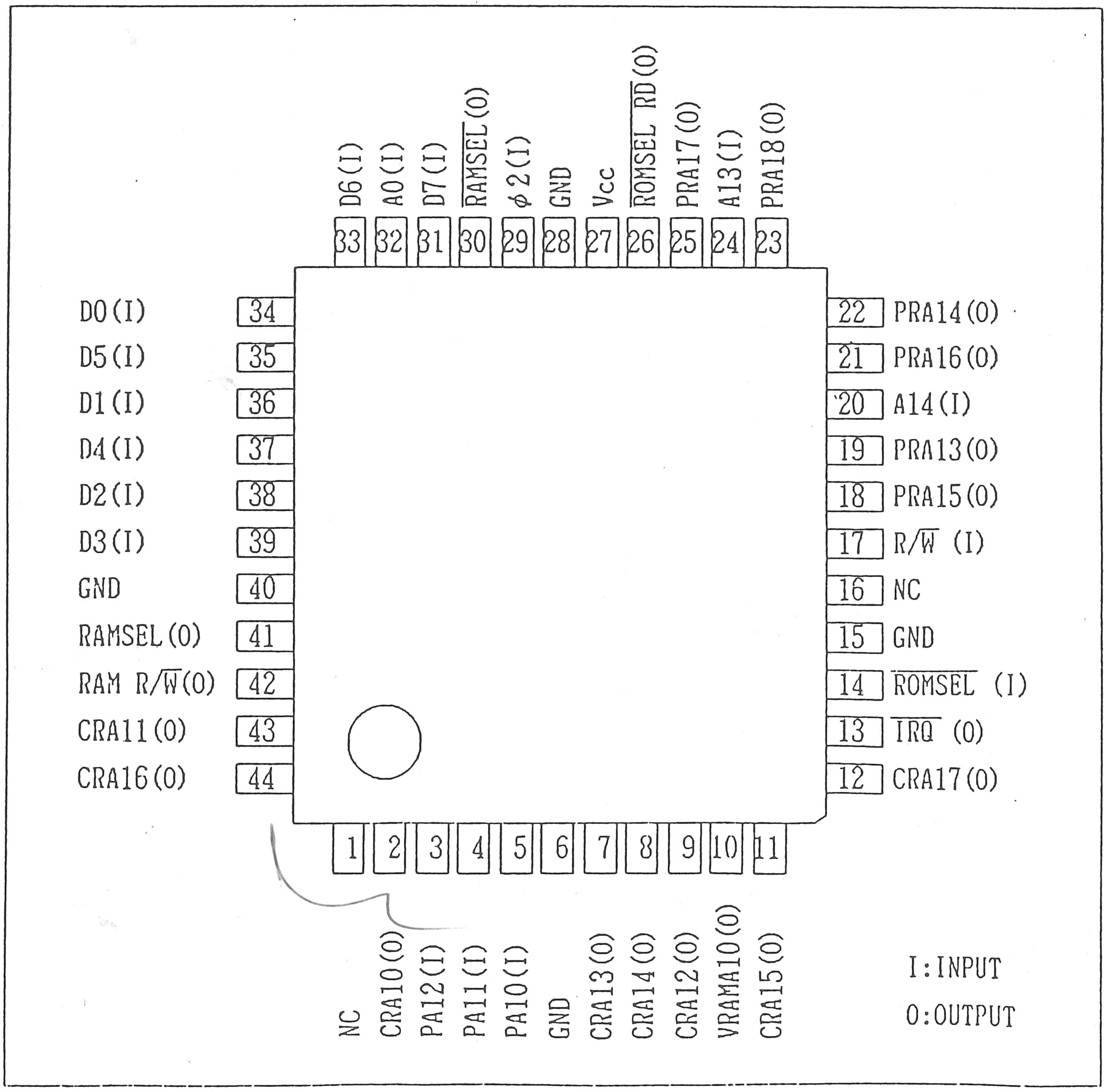
MMC 3 仕様書(第三版)

(Multi-Memory Controller 3)

1988年 5月24日

<概 要>

- ﾌﾟﾛｸﾞﾗﾑ ROM : 4M bit(512K Byte)まで拡張可能。
  CPU ROM Addr. area 8000H-FFFFH のうち、ソフトで
  Home area/Window area 組合せ 2 Mode 設定可能。
  2つの Window area から BANK 拡張のための独立した
  2 BANK REGISTER 内蔵。(R6,R7)
- ｷｬﾗｸﾀｰ ROM : 2M bit(256K Byte)まで拡張可能。
  64CHR./128CHR.単位で BANK 拡張のための独立した 6 BANK
  REGISTER 内蔵。(R0,R1,R2,R3,R4,R5)
- Scroll Switch : ソフトで VRAM を H/V Scroll 対応に切替可能。
- RAMSEL出力 : CPU Addr. 6000H ～7FFFH RAM area用 CS 出力と $\overline{CS}$ 出力
  Back-Up 用に使用の際のソフト Protect付。
- Interval Timer : 多様なソフトに対応する IRQ 発生用 Programmable Interval
  Timer 内蔵。(ｸﾛｯｸT＝1H)



端子配列図 (TOP VIEW)

＜詳細説明＞

1．ＣＰＵからのＭＭＣ３内部 REGISTERの Addr.及び DATA Bit 対応

Register	MSB							LSB	Addr.
POINTER	d7	d6				d2	d1	d0	8000H WR

- d6 → Map Mode
  - Mode 0 : L
  - Mode 1 : H
- d7 → Chr.Seg.
  - Interval Timerを使用しないGameの場合 : L
  - Interval Timerを使用するGameの場合 : L／H 別項参照
- d2〜d0 → R0〜R7指定 → 8001H WR

Register									Addr.
4-5 R0 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1		R0指定後 8001H WR
6-7 R1 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1		R1指定後 8001H WR
0 R2 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R2指定後 8001H WR
1 R3 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R3指定後 8001H WR
2 R4 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R4指定後 8001H WR
3 R5 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R5指定後 8001H WR
R6 : プログラム ROM BANK			d5	d4	d3	d2	d1	d0	R6指定後 8001H WR
R7 : プログラム ROM BANK			d5	d4	d3	d2	d1	d0	R7指定後 8001H WR

注 : R0〜R7の指定はPOINTER のd0〜d2 Bitによって行われる。

Register			Addr.
Scroll SW : ｽｸﾛｰﾙ切替		d0	A000H WR
Protect : Back-Up RAM用	d7	d6	A001H WR

- Scroll SW d0: H Scroll : L / V Scroll : H
- Protect d6 → WR PROTECT
- Protect d7 → $\overline{\mathrm{RAMSELMASK}}$

Register									Addr.
JAM : Interval Timer用	d7	d6	d5	d4	d3	d2	d1	d0	C000H WR

Preset & Start Interval Timer —— P&SIT ——→ C001H WR Soft Switch

DISABLE Interval Timer IRQ —— DITIRQ ——→ E000H WR Soft Switch

ENABLE Interval Timer IRQ —— EITIRQ ——→ E001H WR Soft Switch

2. プログラム ROM BANK (Map Mode 0)

CPU 側の Address(READ)			
A15	A14	A13	A12 ～ A0
H	H	H	―――
H	H	L	―――
H	L	H	―――
H	L	L	―――

Address	
FFFFH ～ E000H	固定 Home
E000H ～ C000H	Home
C000H ～ A000H	Window 1 ⇒
A000H ～ 8000H	Window 0 ⇒

プログラム ROM 側の Address						
PRA18	PRA17	PRA16	PRA15	PRA14	PRA13	PRA12 ～ PRA 0
H	H	H	H	H	H	CPU A12 ～ A0 そのまま
H	H	H	H	H	L	
d5 ――― R7(6Bit) ――― d0						
d5 ――― R6(6Bit) ――― d0						

プログラム ROM BANK (Map Mode 1)

CPU 側の Address(READ)			
A15	A14	A13	A12 ～ A0
H	H	H	―――
H	H	L	―――
H	L	H	―――
H	L	L	―――

Address	
FFFFH ～ E000H	固定 Home
E000H ～ C000H	Window 0 ⇒
C000H ～ A000H	Window 1 ⇒
A000H ～ 8000H	Home

プログラム ROM 側の Address						
PRA18	PRA17	PRA16	PRA15	PRA14	PRA13	PRA12 ～ PRA 0
H	H	H	H	H	H	CPU A12 ～ A0 そのまま
d5 ――― R6(6Bit) ――― d0						
d5 ――― R7(6Bit) ――― d0						
H	H	H	H	H	L	
				↑	↑	

要注意

注1. 例えば、Prog. Counter が A000H ～ BFFFH にあって、R7を書き換えた場合、STA 8001H 実行後次の命令からは書き換えられた BANK となる。

3. キャラクター ROM BANK

PPU からの Address						キャラクター ROM 側の Address		
PA12	PA11	PA10	PA9~PA0	→		CRA17 16 15 14 13 12 11	10	CRA9 ~CRA0
L	L	—	——	0000H	CHR.No0 ~ CHR.No7F	d7 —— R0(7Bit) —— d1	PPU PA10~PA0 そのまま	
L	H	—	——	0800H	CHR.No80 ~ CHR.NoFF	d7 —— R1(7Bit) —— d1		
H	L	L	——	1000H	CHR.No0 ~ CHR.No3F	d7 —— R2(8Bit) —— d0		PPU PA9 ~PA0 そのまま
H	L	H	——	1400H	CHR.No40 ~ CHR.No7F	d7 —— R3(8Bit) —— d0		
H	H	L	——	1800H	CHR.No80 ~ CHR.NoBF	d7 —— R4(8Bit) —— d0		
H	H	H	——	1C00H	CHR.NoC0 ~ CHR.NoFF	d7 —— R5(8Bit) —— d0		
				1FFFH				

注1 Interval Timerを全く使用しないGameの場合で、MMC3 内部POINTER にDATAを SETする際は、Chr.Seg.Bit(d7)を常に“L”に保持させること。そうすることによって、PPU 内部のOBJ.Seg.Bit, BG.Seg.Bitの状態に応じて0000H～0FFFH、1000H～1FFFHとそれぞれ OBJ., BG.領域を従来通り割り付けることができる。

OBJ. 8×16 pixelについても従来通りの対応となる。

注2 Interval Timerを使用するGameの場合は、PPU 内部のOBJ.Seg.Bit を“H”、BG.Seg.Bitを“L”に固定して使用すること。すると、MMC3 内部POINTER のChr.Seg.Bit(d7) に応じて、PPU からの Addressを下表のように割り付けることができる。

	PPU からの Address	
Chr. Seg.(d7)	OBJ.側	BG.側
L	1000H ～ 1FFFH (R2～R5)	0000H ～ 0FFFH (R0～R1)
H	0000H ～ 0FFFH (R0～R1)	1000H ～ 1FFFH (R2～R5)

又この時、OBJ.を 8×16 pixelで使用する時も上の表の Address割り付けに従うが、OBJ.の CHR.Noは必ず奇数番号で使用すること!!

但し、BG. に関してはこの限りではない。

注3 プログラムROM 、キャラクターROM 用 BANK R0～R7を指定するためにPOINTER を SETする際は、Chr.Seg.& Map Mode Bitが共存しているので要注意!!

4. Ｓｃｒｏｌｌ切替

H Scroll 対応 (VRAMA10＝PA10) : LDA #00H & STA A000H
V Scroll 対応 (VRAMA10＝PA11) : LDA #01H & STA A000H

上記のように一度 A000H番地に Scroll 対応をセットすれば、その状態は
再度 A000H番地のDATA書換えまでは保持される。

5. ＲＡＭＳＥＬ出力　Address 6000H～7FFFHを RAM領域に設定するための正論理及び負論理出力

RAMSEL (READ時) ＝ (6000H～7FFFH READ) で $\overline{\mathrm{RAMSELMASK}}$＝H WRPROTECT＝Any
RAMSEL(Write時) ＝ (6000H～7FFFH Write)で $\overline{\mathrm{RAMSELMASK}}$＝H 且つ WRPROTECT＝L

特にこの領域を Battery-Back upする場合は、以下の注意が必要。

Ａ. Game終了する場合は、Save動作でない時 RESETを押しながら POWER OFFする
注意ラベルを付ける。

Ｂ. $\overline{\mathrm{RAMSELMASK}}$、WRPROTECT BIT で必要な時のみ RAMを OPENし、確率的にミスを
少なくする。

Ｃ. Saveは、できるだけ二重の Address領域にDATAを Writeして安全性を高める。
又は、Error Correctionを導入する。

Ｄ. Battery-Back up をした場合、POWER OFF時 6000H,6001H,7000H,7001H番地のDATAは
壊れやすいので、この４つの番地はSave領域からはずすこと。

上のＡ～Ｄの組み合せにより、Back-up Saveの信頼性を向上させることができる。

6. Interval Timer IRQ (以下 ITIRQ と略)

Ａ. ITIRQ も CPU LSI内部 IRQもどちらも使用しないGameの場合

CPU RESET で CPUの INTERRUPT DISABLE Flag は SETされるので、ソフトで
CLI 命令を使用しなければよい。（念のため、ソフトでも RESET時 SEI命令を
入れておくのが望ましい）
又項目３で述べたように、MMC3 内部 POINTERの Chr.Seg.Bitは常に "L"を
保持させて使用する必要がある。

Ｂ. CPU LSI 内部 IRQのみ使用し、ITIRQは使用しないGameの場合

CPU RESET後、CLI 命令までの間に DITIRQ(E000H) WR を１度だけ実行して
おいて、それ以後 EITIRQ(E001H) WR を実行しない。このことにより、IRQ
発生源は CPU LSI内部からのみとして処理できる。
又項目３で述べたように、MMC3 内部 POINTERの Chr.Seg.Bitは常に "L"を
保持させて使用する必要がある。

C．ITIRQを使用するGameの場合

（場面により ITIRQを発生させたり、させなかったりするGameもこれにふくまれる。又 ITIRQ、CPU LSI 内部 IRQを同時に使用することは不可能であるが、場面により使い分けることは原理的には可能である。しかし、現実的にはCPU LSI 内部 IRQを使用することはほとんどなく、そのような両方使用する場合はないものとする。）

CPU RESET 時の Initialization

1．NMI REQ． 発生禁止

PPU 内部 CTLR0(2000H)　OBJ.Seg.Bit ＝ H, BG.Seg.Bit ＝ L,
NMI ENABLE Bit ＝ L　WR

OBJ.Seg., BG.Seg.Bitは前にも述べたように ITIRQを使用するGameにおいては常にそれぞれ OBJ.Seg.Bit "H", BG.Seg.Bit "L"を保持させる必要がある。

2．SEI 命令

Set Interrupt DISABLE Flag

3．Back-Up Protect (必要ならば)

Protect(A001H)に 40H WR

4．画面BLANK

CTLR1(2001H)の OBJ. と BG. Blanking Bit 両方ともに "H" を Setする。

5．Sound 関係 REGISTER Initialization

この時 Sound WRE0(4010H)d7 Bit L, SOFTCLK(4017H)40H WR も同時に行い、CPU LSI 内部 IRQ発生を禁止させる。

特に WRE0 d7 Bitに関しては、後で他の Bit使用の際も "L" を保持させる必要がある。

6．Interval Timer Initialization

DITIRQ(E000H) WR

1H clock Initialize →

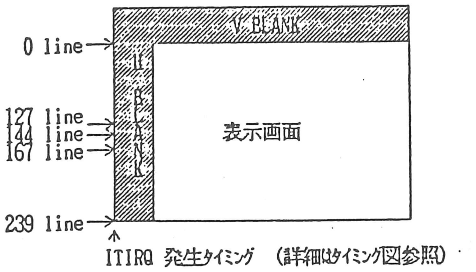
```
    LDA  2002H
    LDA  #10H
    TAX
LP  STA  2006H
    STA  2006H
    EOR  #10H
    DEX
    BNE  LP
```

———— ・ ————

以下、Video 画面上に0 lineから 239 lineまで表示されていると考え、127 lineの画面表示中に Service 1 を実行し、144 lineの H.BLANK中に Service 2 を実行し、さらに 167 lineの画面表示中に Service 3 を実行するものとし、それを 1 Field毎に繰り返す例を説明する。

Interval Timer の使用例

あくまでも原理説明のためのソフト例であるので、実際は工夫が必要です。



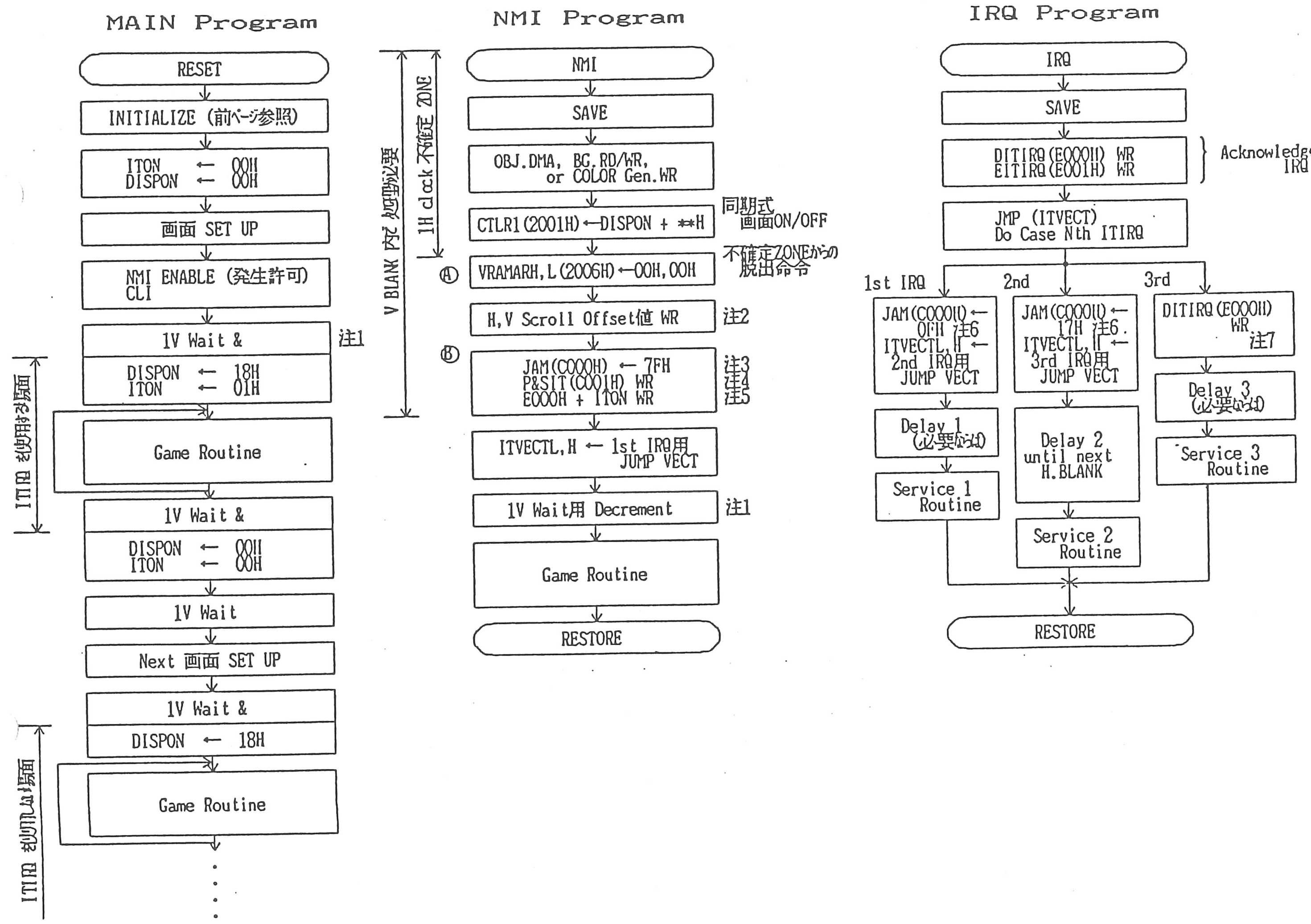
Working RAM 内に以下の Addressを EQU. 文にて設定する。

DISPON = 画面 ON/OFF 用 Flag 18H=ON, 00H=OFF

ITON = ITIRQ を使用する場面かどうかの Flag 00H=使用しない場面, 01H=使用する場面

連続 { ITVECTL = Do Case Nth ITIRQ 用 JUMP Address Low

ITVECTH = Do Case Nth ITIRQ 用 JUMP Address High



注1. 1V Wait は、NMI Program から EXIT した Timing と同期化させる目的のもので、画面 ON/OFF や ITIRQ関係をうまく処理するために MAIN Programで 01Hを Working RAMのある番地に SETし、NMI Program 内の Decrement Routine結果で 00Hになるのを Check & Wait する MAIN Program 側の Sub-Routineをこの使用例では想定している。

注2. CTLR0 (2000H) WR で Scroll H,V の MSBを設定の際は、OBJ. Seg."H", BG. Seg."L" を忘れずに!!

注3. 1st IRQ は 127 line の頭で必要だから 7FHを JAMに SET。一般的に V BLANK内の P&SITの前の JAM設定値は、N lineの頭で IRQを必要とするならば、N の Hex. 値で最大は F0Hである。00H も可能である。

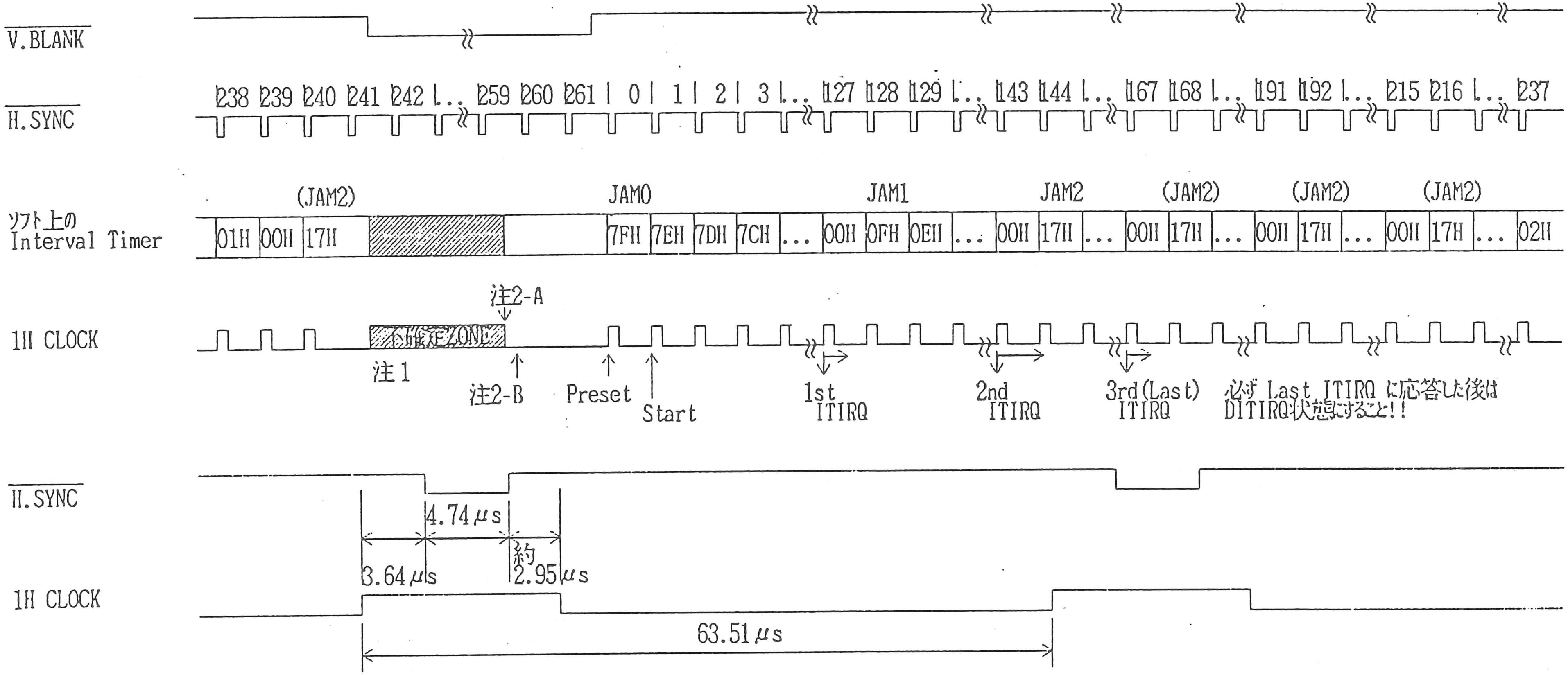
注4. P&SIT 命令は、V BLANK 内で毎 field 一度だけ必要であり、JAM1,JAM2 の Preset は自動的に実行される。

注5. ITONの値 (00H/01H) に応じて DITIRQ or EITIRQ となる。又この命令が実行されるまでは必ず DITIRQ の状態であること!!

注6. 2nd IRQ は 143 line で必要とし、Delay 2 で144 lineの頭の H Blanking 内まで Delayさせるから 0FHを JAMに SETする。又、3rd IRQ は 167 line の頭で必要とするから 17Hを JAMに SETする。一般的に 2nd,3rd IRQ用の JAMの設定値は、N lineの頭で前の IRQがあり、M lineで次の IRQが必要とするならば (M-N+1)の Hex. 値である。但し、00H は禁止である。

注7. 3rd IRQ (Last ITIRQ)では、必ず DITIRQ の状態にしておくこと!!

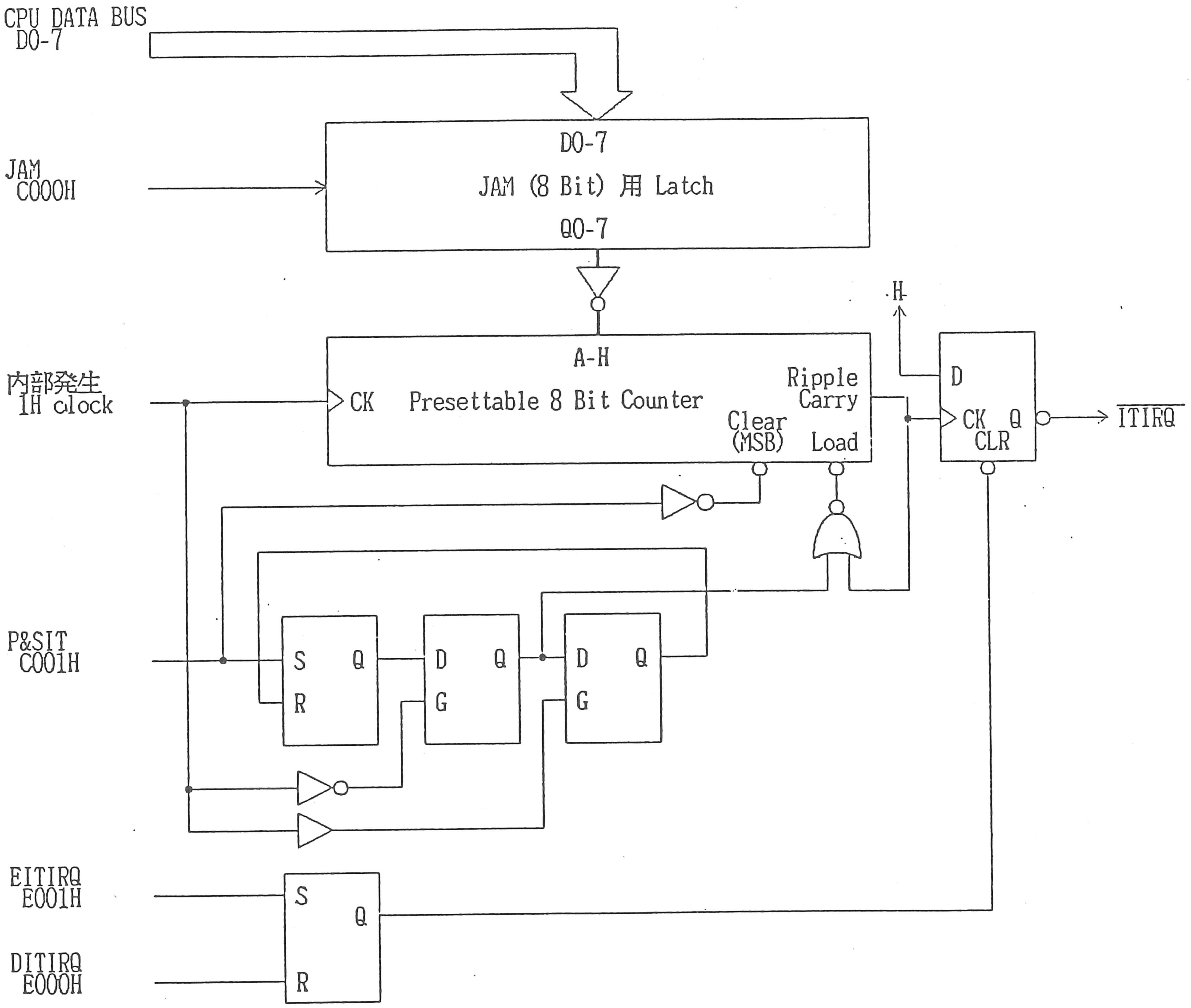
Interval Timer タイミング図



注1　CPU から PPUを経由して VRAM, COLOR Gen. CHR. ROM/RAM を ACCESS するためには、それぞれの Addressを PPU内部 REGISTER に SETする必要がある。その SETする Addess によって信号レベルが決定されるので、一応不確定ZONEとして考えることにする。しかしこのZONEの時間の長さはソフトに依存するもので、ソフト上コントロール可能である。
関連ではあるが、PPU 内部の OBJ. BLANKING, BG. BLANKING Bitによって OBJ. BG. 両方とも Blanking させた時は、1H CLOCK は常時不確定ZONEとなる。もし、Interval Timerを使用する場面で OBJ. BG. 両方とも全面 ON/OFF させたい時は、COLOR Gen. による輝度変化、又は片方だけ交互に Blankingさせる等の代替手段で対処すること。

注2-A, 2-B　それぞれ Interval Timer 使用例の Ⓐ, Ⓑ に相当するので参照。

MMC内部の Interval Timer 等価回路



注) Interval Timerはハード的には Up Counter であり、FFH で Ripple Carry を発生する。
しかし、JAM から Preset の際は 1's Complement で実行されるので、ソフト上は JAMの
値から 00Hまでの 8 Bit Down Count Timer として考られる。

暫定

ＭＭＣ３ 仕様書（案）

(Multi-Memory Controller 3)

1987年12月15日

<概 要>

- ・ﾌﾟﾛｸﾞﾗﾑ ROM ： 4M bit(512K Byte)まで拡張可能。
  CPU ROM Addr. area 8000H-FFFFH のうち、ソフトで
  Home area／Window area 組合せ２Mode 設定可能。
  ２つの Window area から BANK 拡張のための独立した
  2 BANK REGISTER 内蔵。(R6,R7)
- ・ｷｬﾗｸﾀｰ ROM ： 2M bit(256K Byte)まで拡張可能。
  64CHR./128CHR.単位で BANK 拡張のための独立した 6 BANK
  REGISTER 内蔵。(R0,R1,R2,R3,R4,R5)
- ・Scroll Switch ： ソフトで VRAM を H／V Scroll 対応に切替可能。
- ・RAMSEL出力 ： CPU Addr. 6000H ～7FFFH RAM area用 CS 出力と $\overline{\text{CS}}$ 出力
  Back-Up 用に使用の際のソフト Protect付。
- ・Interval Timer ： 多様なソフトに対応する IRQ 発生用 Programmable Interval
  Timer 内蔵。(ｸﾛｯｸT＝1H)

φ2(I)	1	42	Vcc
$\overline{\text{RAMSEL}}$(O)	2	41	$\overline{\text{ROMSELRD}}$(O)
D7(I)	3	40	PRA17(O)
A0(I)	4	39	A13(I)
D6(I)	5	38	PRA18(O)
D0(I)	6	37	PRA14(O)
D5(I)	7	36	PRA16(O)
D1(I)	8	35	A14(I)
D4(I)	9	34	PRA13(O)
D2(I)	10	33	PRA15(O)
D3(I)	11	32	R/$\overline{\text{W}}$(I)
GND	12	31	DLYED φ2(O)
RAMSEL(O)	13	30	GND
RAM R/$\overline{\text{W}}$(O)	14	29	$\overline{\text{ROMSEL}}$(I)
CRA11(O)	15	28	$\overline{\text{IRQ}}$(O)
CRA16(O)	16	27	CRA17(O)
CRA10(O)	17	26	CRA15(O)
PA12(I)	18	25	VRAMA10(O)
PA11(I)	19	24	CRA12(O)
PA10(I)	20	23	CRA14(O)
GND	21	22	CRA13(O)

I:INPUT
O:OUTPUT

端子配列図（TOP VIEW）

＜詳細説明＞

1．ＣＰＵからのＭＭＣ３内部 REGISTERの Addr.及び DATA Bit 対応

POINTER: MSB d7 d6 … d2 d1 d0 LSB — 8000H WR

- d7 → Chr.Seg.
  - Interval Timerを使用しないGameの場合 : L
  - Interval Timerを使用するGameの場合 : L／H 別項参照
- d6 → Map Mode
  - Mode 0 : L
  - Mode 1 : H
- d2〜d0: R0〜R7指定

Register	d7	d6	d5	d4	d3	d2	d1	d0	Address
R0 : CHR. BANK		d6	d5	d4	d3	d2	d1	d0	R0指定後 8001H WR
R1 : CHR. BANK		d6	d5	d4	d3	d2	d1	d0	R1指定後 8001H WR
R2 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R2指定後 8001H WR
R3 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R3指定後 8001H WR
R4 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R4指定後 8001H WR
R5 : CHR. BANK	d7	d6	d5	d4	d3	d2	d1	d0	R5指定後 8001H WR
R6 : ﾌﾟﾛｸﾞﾗﾑ ROM BANK			d5	d4	d3	d2	d1	d0	R6指定後 8001H WR
R7 : ﾌﾟﾛｸﾞﾗﾑ ROM BANK			d5	d4	d3	d2	d1	d0	R7指定後 8001H WR

注：R0〜R7の指定はPOINTER のd0〜d2 Bitによって行われる。

Scroll SW ：ｽｸﾛｰﾙ切替 — d0 — A000H WR

- H Scroll : L
- V Scroll : H

Protect : Back-Up RAM用 — d7 d6 — A001H WR

- d7 → RAMSELMASK
- d6 → WR PROTECT

Register	d7	d6	d5	d4	d3	d2	d1	d0	Address
JAM : Interval Timer用	d7	d6	d5	d4	d3	d2	d1	d0	C000H WR

Preset & Start Interval Timer —— P&SIT ——→ C001H WR Soft Switch

DISABLE Interval Timer IRQ —— DITIRQ ——→ E000H WR Soft Switch

ENABLE Interval Timer IRQ —— EITIRQ ——→ E001H WR Soft Switch

2. プログラム ROM BANK (Map Mode 0)

CPU 側の Address(READ)						プログラム ROM 側の Address						
A15	A14	A13	A12 ～ A0	(CPU address)		PRA18	PRA17	PRA16	PRA15	PRA14	PRA13	PRA12 ～ PRA 0
H	H	H	――――	FFFF–E000	固定 Home	H	H	H	H	H	H	CPU A12 ～ A0 そのまま
H	H	L	――――	E000–C000	Home	H	H	H	H	H	L	CPU A12 ～ A0 そのまま
H	L	H	――――	C000–A000	Window 1 ⇒	d5 ――― R7(6Bit) ――― d0						CPU A12 ～ A0 そのまま
H	L	L	――――	A000–8000	Window 0 ⇒	d5 ――― R6(6Bit) ――― d0						CPU A12 ～ A0 そのまま

プログラム ROM BANK (Map Mode 1)

CPU 側の Address(READ)						プログラム ROM 側の Address						
A15	A14	A13	A12 ～ A0	(CPU address)		PRA18	PRA17	PRA16	PRA15	PRA14	PRA13	PRA12 ～ PRA 0
H	H	H	――――	FFFF–E000	固定 Home	H	H	H	H	H	H	CPU A12 ～ A0 そのまま
H	H	L	――――	E000–C000	Window 0 ⇒	d5 ――― R6(6Bit) ――― d0						CPU A12 ～ A0 そのまま
H	L	H	――――	C000–A000	Window 1 ⇒	d5 ――― R7(6Bit) ――― d0						CPU A12 ～ A0 そのまま
H	L	L	――――	A000–8000	Home	H	H	H	H	H	L	CPU A12 ～ A0 そのまま
										↑	↑	
										要注意		

注１．　例えば、Prog. Counter が A000H ～ BFFFH にあって、Ｒ７を書き換えた場合、STA 8001H 実行後次の命令からは書き換えられた BANK となる。

3. キャラクター ROM BANK

PPU からの Address				→		キャラクター ROM 側の Address		
PA12	PA11	PA10	PA9~PA0			CRA17 16 15 14 13 12 11	10	CRA9 ~CRA0
L	L	—	——	0000	CHR.No0 〜 CHR.No7F	d6 ——— R0(7Bit) ——— d0	PPU PA10〜PA0 そのまま	
L	H	—	——	0800	CHR.No80 〜 CHR.NoFF	d6 ——— R1(7Bit) ——— d0		
H	L	L	——	1000	CHR.No0 〜 CHR.No3F	d7 ——— R2(8Bit) ——— d0		PPU PA9 〜PA0 そのまま
H	L	H	——	1400	CHR.No40 〜 CHR.No7F	d7 ——— R3(8Bit) ——— d0		
H	H	L	——	1800	CHR.No80 〜 CHR.NoBF	d7 ——— R4(8Bit) ——— d0		
H	H	H	——	1C00	CHR.NoC0 〜 CHR.NoFF	d7 ——— R5(8Bit) ——— d0		
				1FFF				

注1 Interval Timerを全く使用しないGameの場合で、MMC3 内部POINTER にDATAを SETする際は、Chr.Seg.Bit(d7)を常に“L”に保持させること。そうすることによって、PPU 内部のOBJ.Seg.Bit, BG.Seg.Bitの状態に応じて0000〜0FFF、1000〜1FFFとそれぞれ OBJ., BG.領域を従来通り割り付けることができる。
OBJ. 8×16 pixelについても従来通りの対応となる。

注2 Interval Timerを使用するGameの場合は、PPU 内部のOBJ.Seg.Bit を“H”、BG.Seg.Bitを“L”に固定して使用すること。すると、MMC3 内部POINTER のChr.Seg.Bit(d7) に応じて、PPU からの Addressを下表のように割り付けることができる。

	PPU からの Address	
Chr. Seg.(d7)	OBJ.側	BG.側
L	1000 〜 1FFF (R2〜R5)	0000 〜 0FFF (R0〜R1)
H	0000 〜 0FFF (R0〜R1)	1000 〜 1FFF (R2〜R5)

又この時、OBJ.を 8×16 pixelで使用する時も上の表の Address割り付けに従うが、OBJ.の CHR.Noは必ず奇数番号で使用すること!!
但し、BG. に関してはこの限りではない。

注3 プログラムROM 、キャラクターROM 用 BANK R0〜R7を指定するためにPOINTER を SETする際は、Chr.Seg.& Map Mode Bitが共存しているので要注意!!

4. Ｓｃｒｏｌｌ切替

H Scroll 対応 (VRAM10＝PA10) : LDA #00H & STA A000H
V Scroll 対応 (VRAM10＝PA11) : LDA #01H & STA A000H

上記のように一度 A000 番地に Scroll 対応をセットすれば、その状態は再度 A000番地のDATA書換えまでは保持される。

5. ＲＡＭＳＥＬ出力 Address 6000～7FFFを RAM領域に設定するための正論理及び負論理出力

RAMSEL (READ時) ＝ (6000～7FFF READ) ・ $\overline{\text{RAMSELMASK}}$
RAMSEL(Write時) ＝ (6000～7FFF Write) ・ $\overline{\text{RAMSELMASK}}$ ・ $\overline{\text{WRPROTECT}}$

特にこの領域を Battery-Back upする場合は、以下の注意が必要。

A. Game終了する場合は、Save動作でない時 RESETを押しながら POWER OFFする注意ラベルを付ける。
B. $\overline{\text{RAMSELMASK}}$、$\overline{\text{WRPROTECT}}$ BIT で必要な時のみ RAMを OPENし、確率的にミスを少なくする。
C. Saveは、できるだけ二重の Address領域にDATAを Writeして安全性を高める。又は、Error Correctionを導入する。
D. Battery-Back up をした場合、POWER OFF時 6000H,7000H番地のDATAは壊れやすいので、この２つの番地はSave領域からはずすこと。

上のＡ～Ｄの組み合せにより、Back-up Saveの信頼性を向上させることができる。

6. Interval Timer IRQ (以下 ITIRQ と略)

A. ITIRQ も CPU LSI内部 IRQもどちらも使用しないGameの場合
CPU RESET で CPUの INTERRUPT DISABLE Flag は SETされるので、ソフトで CLI 命令を使用しなければよい。(念のため、ソフトでも RESET時 SEI命令を入れておくのが望ましい)
又項目３で述べたように、MMC3 内部 POINTERの Chr.Seg.Bitは常に“L”を保持させて使用する必要がある。

B. CPU LSI 内部 IRQのみ使用し、ITIRQは使用しないGameの場合
CPU RESET後、CLI 命令までの間に DITIRQ(E000H) WR を１度だけ実行しておいて、それ以後 EITIRQ(E001H) WR を実行しない。このことにより、IRQ発生源は CPU LSI内部からのみとして処理できる。
又項目３で述べたように、MMC3 内部 POINTERの Chr.Seg.Bitは常に“L”を保持させて使用する必要がある。

C．ITIRQを使用するGameの場合

（場面により ITIRQを発生させたり、させなかったりするGameもこれにふくまれる。又 ITIRQ、CPU LSI 内部 IRQを同時に使用することは不可能であるが、場面により使い分けることは原理的には可能である。しかし、現実的にはCPU LSI 内部 IRQを使用することはほとんどなく、そのような両方使用する場合はないものとする。）

CPU RESET 時の Initialization

1．NMI REQ. 発生禁止

PPU 内部 CTLR0(2000H)　OBJ.Seg.Bit = H, BG.Seg.Bit = L,
NMI ENABLE Bit = L WR

OBJ.Seg., BG.Seg.Bitは前にも述べたように ITIRQを使用するGameにおいては常にそれぞれ OBJ.Seg.Bit "H", BG.Seg.Bit "L"を保持させる必要がある。

2．SEI 命令

Set Interrupt DISABLE Flag

3．Back-Up Protect（必要ならば）

Protect(A001H)に 00H WR

4．画面BLANK

CTLR1(2001H)の OBJ. と BG. Blanking Bit 両方ともに "H" を Setする。

5．Sound 関係 REGISTER Initialization

この時 Sound WRE0(4010H)d7 Bit L, SOFTCLK(4017H)40H WR も同時に行い、CPU LSI 内部 IRQ発生を禁止させる。
特に WRE0 d7 Bitに関しては、後で他の Bit使用の際も "L" を保持させる必要がある。

6．Interval Timer Initialization

DITIRQ(E000H) WR

1H clock Initialize ──────→

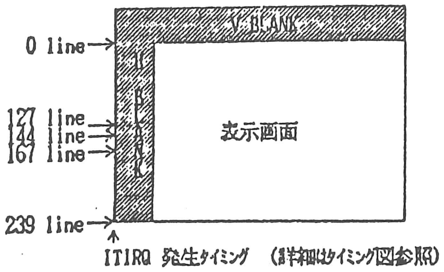
```
     LDA   2002H
     LDA   #10H
     TAX
LP   STA   2006H
     STA   2006H
     EOR   #00H
     DEX
     BNE   LP
```

───── ・ ─────

以下、Video 画面上に0 lineから 239 lineまで表示されていると考え、127 lineの画面表示中に Srvice 1 を実行し、144 lineの H.BLANK中に Service 2 を実行し、さらに 167 lineの画面表示中に Service 3 を実行するものとし、それを 1 Field毎に繰り返す例を説明する。

Interval Timer の使用例

あくまでも原理説明のためのソフト例であるので、実際は工夫が必要です。

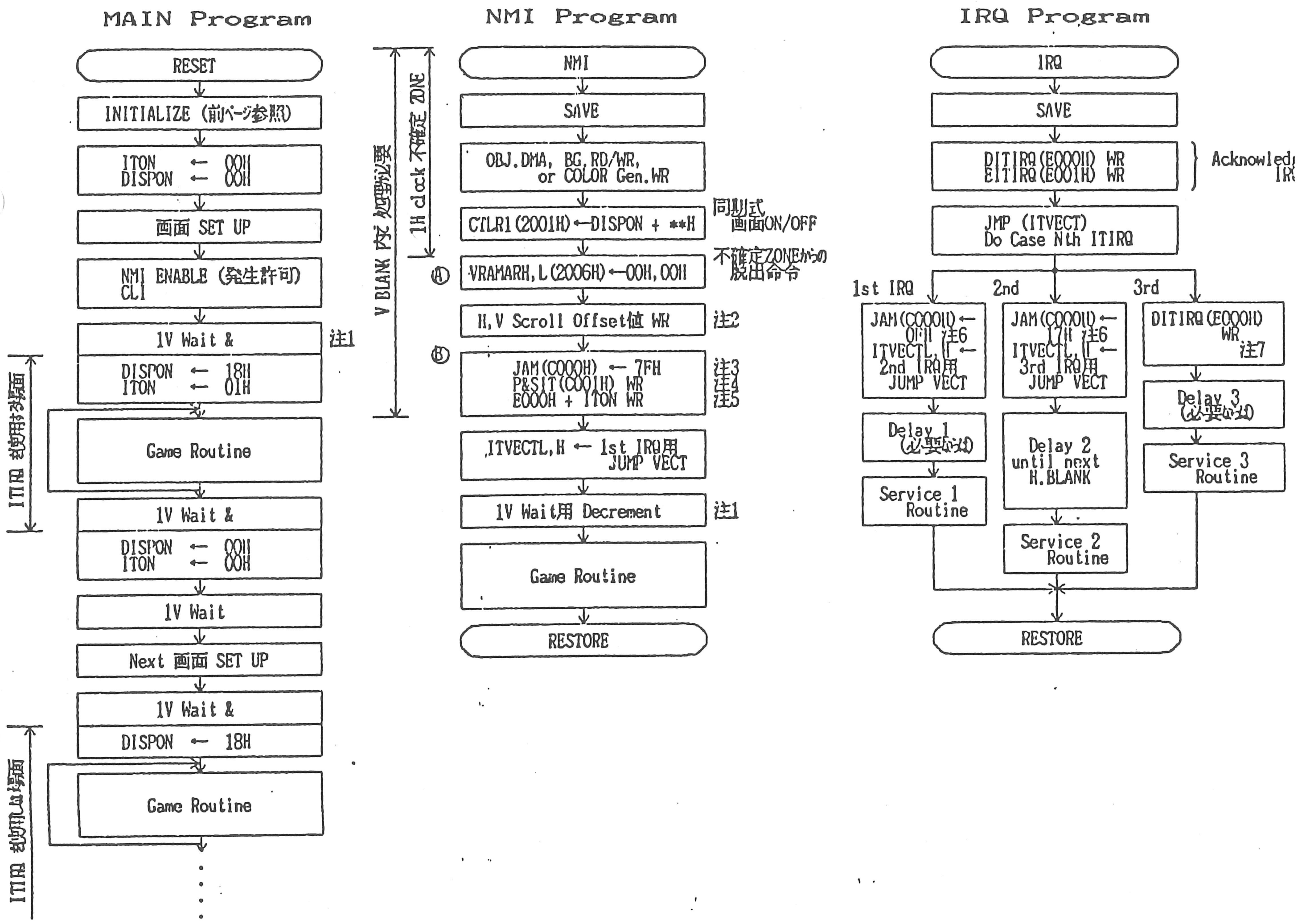


Working RAM 内に以下の Addressを EQU. 文にて設定する。

DISPON = 画面 ON/OFF 用 Flag 18H=ON, 00H=OFF

ITON = ITIRQ を使用する場面かどうかの Flag 00H=使用しない場面, 01H=使用する場面

連続 { ITVECTL = Do Case Nth ITIRQ 用 JUMP Address Low
ITVECTH = Do Case Nth ITIRQ 用 JUMP Address High }



注1. 1V Wait は、NMI Program から EXIT した Timing と同期化させる目的のもので、画面 ON/OFF や ITIRQ関係をうまく処理するために MAIN Programで 01Hを Working RAMのある番地に SETし、NMI Program 内の Decrement Routine結果で 00Hになるのを Check & Wait する MAIN Program 側の Sub-Routineをこの使用例では想定している。

注2. CTLR0(2000H) WR で Scroll H,V の MSBを設定の際は、OBJ. Seg."H", BG. Seg."L" を忘れずに!!

注3. 1st IRQ は 127 line の頭で必要だから 7FHを JAMに SET。一般的に V BLANK内の P&SITの前の JAM設定値は、N lineの頭で IRQを必要とするならば、N の Hex. 値で最大は F0Hである。00H も可能である。

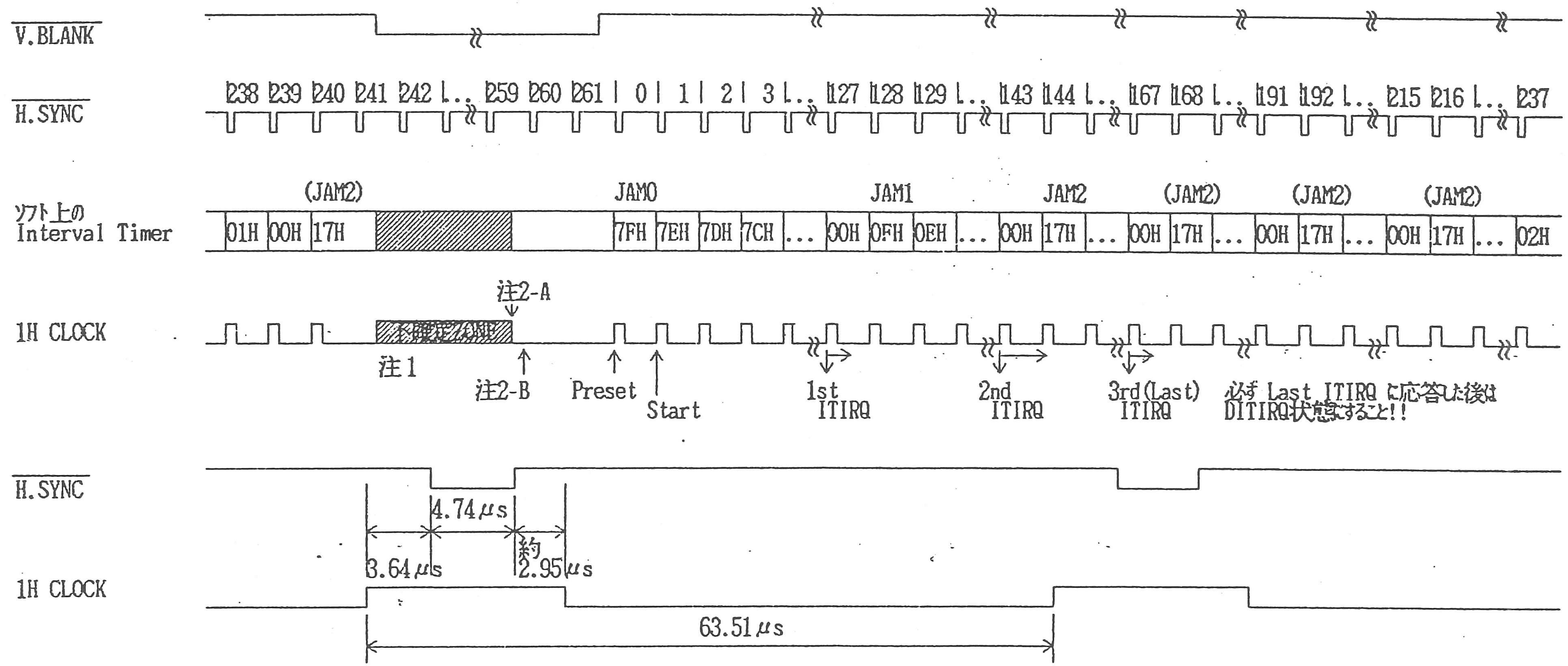
注4. P&SIT 命令は、V BLANK 内で毎 field 一度だけ必要であり、JAM1,JAM2 の Preset は自動的に実行される。

注5. ITONの値(00H/01H) に応じて DITIRQ or EITIRQ となる。又この命令が実行されるまでは必ず DITIRQ の状態であること!!

注6. 2nd IRQ は 143 line で必要とし、Delay 2 で144 lineの頭の H Blanking 内まで Delayさせるから 0FHを JAMに SETする。
又、3rd IRQ は 167 line の頭で必要とするから 17Hを JAMに SETする。一般的に 2nd,3rd IRQ用の JAMの設定値は、N lineの頭で前の IRQがあり、M lineで次の IRQが必要とするならば (M-N+1)の Hex. 値である。但し、00H は禁止である。

注7. 3rd IRQ (Last ITIRQ)では、必ず DITIRQ の状態にしておくこと!!

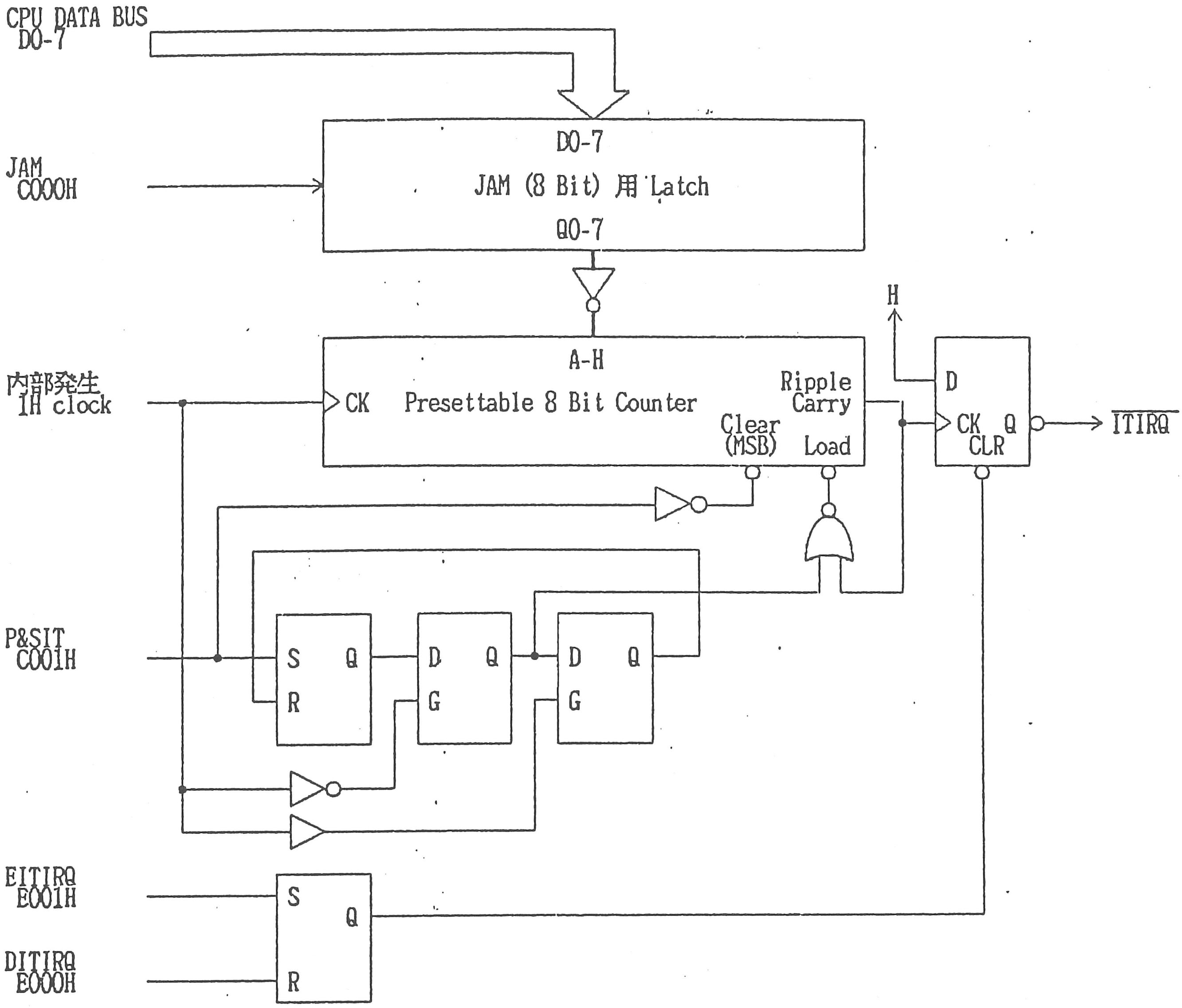
Interval Timer タイミング図



注1　CPU から PPUを経由して VRAM, COLOR Gen. CHR. ROM／RAM を ACCESS するためには、それぞれの Addressを PPU内部 REGISTER に SETする必要がある。その SETする Addess によって信号レベルが決定されるので、一応不確定ZONEとして考えることにする。しかしこのZONEの時間の長さはソフトに依存するもので、ソフト上コントロール可能である。
関連ではあるが、PPU 内部の OBJ. BLANKING, BG. BLANKING Bitによって OBJ. BG. 両方とも Blanking させた時は、1H CLOCK は常時不確定ZONEとなる。もし、Interval Timerを使用する場面で OBJ. BG. 両方とも全面 ON／OFF させたい時は、COLOR Gen. による輝度変化、又は片方だけ交互に Blankingさせる等の代替手段で対処すること。

注2-A,2-B　それぞれ Interval Timer 使用例の Ⓐ, Ⓑ に相当するので参照。

MMC内部の Interval Timer 等価回路



注) Interval Timerはハード的には Up Counter であり、FFH で Ripple Carry を発生する。
しかし、JAM から Preset の際は 1's Complement で実行されるので、ソフト上は JAMの値から 00Hまでの 8 Bit Down Count Timer として考られる。

ＭＭＣ３　ＣＨＥＣＫＥＲ

取扱説明書

任天堂株式会社

概要

ＭＭＣ３　ＣＨＥＣＫＥＲは，ＭＭＣ３対応ＲＯＭカセットのデバッグ及び最終検査時に使用します．使用方法は，ＣＨＥＣＫＥＲの基板上にあるディップスイッチをプログラムサイズ及びキャラクターサイズに対応して設定し所定の位置にＥＰ－ＲＯＭを差し込むだけでＭＭＣ３対応基板と同じ環境でデバッグ及び最終検査が手軽にできます．

１．ディップスイッチ設定表

表－１．２．３を参照の上，基板上のディップスイッチを設定します．

１）ＰＲＧサイズ

	ＳＷ１	ＳＷ２	ＩＣ	実装位置
４Ｍ	ＯＦＦ	ＯＦＦ	マスク対応 1M EP-ROM	U3,U4,U5,U6
２Ｍ	ＯＦＦ	ＯＮ	〃	U3,U4
１Ｍ	ＯＮ	ＯＦＦ	〃	U3

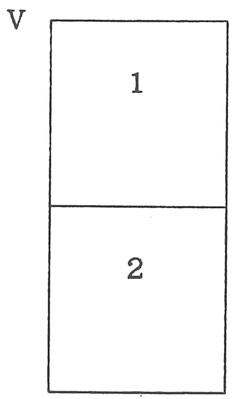
２）ＣＨＲサイズ

	ＳＷ４	ＩＣ	実装位置
２Ｍ	ＯＦＦ	マスク対応 1M EP-ROM	U9,U10
１Ｍ	ＯＮ	〃	U9

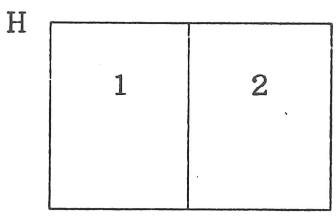
３）ＶＲＡＭ　ＳＷ（ＳＷ３）

	ＳＷ３	
	ＯＦＦ	従来通りＶＲＡＭ２画面モード
	ＯＮ	ＶＲＡＭ４画面モード（追加オプションメモリ）

ＳＷ３　ＯＦＦ時



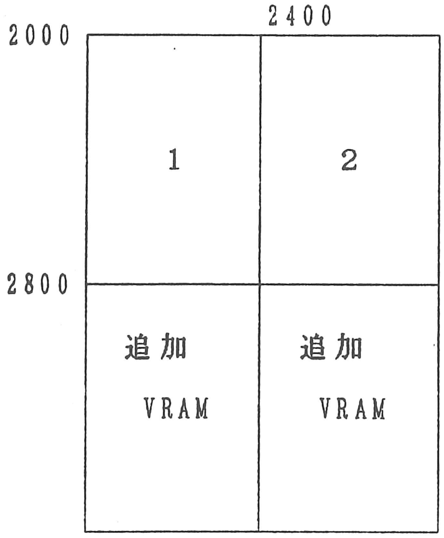
または



（ＭＭＣ３内部スクロール切り替え）

＊　ファミリーコンピュータ本体内ＶＲＡＭ使用

ＳＷ３　ＯＮ時（追加オプションメモリ使用時）



＊ＳＷ３　ＯＮで使用の場合
追加ＶＲＡＭのアドレスが固定されているためＭＭＣ３内部スクロール切り替えは，Ｈで使用の事

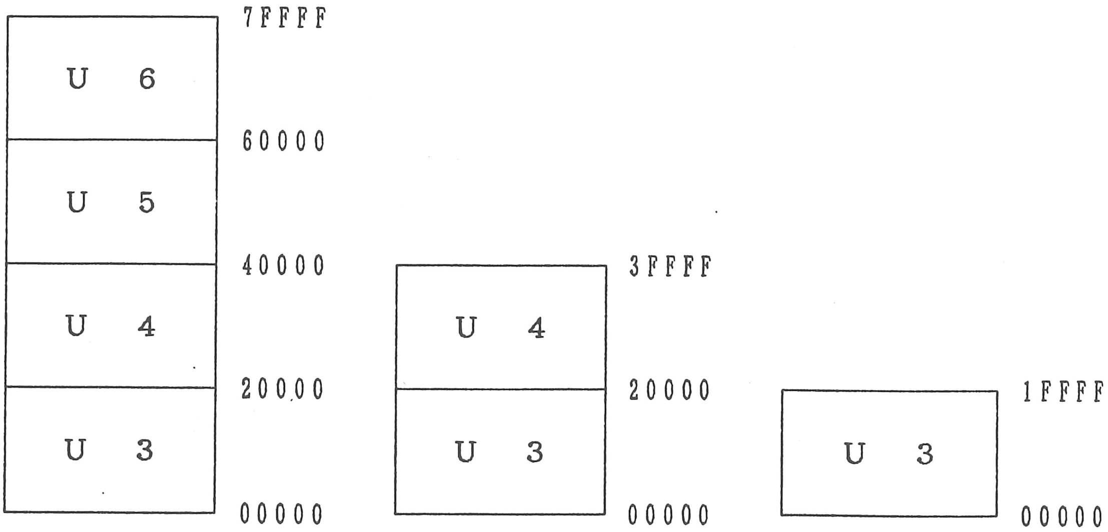
＊１Ｍ　ＥＰ－ＲＯＭは，３２ピンマスク対応ＲＯＭを使用

＊バックアップ仕様の時Ｕ７はＣＭＯＳ６４ＫＢｉｔＲＡＭを使用し，リチウム電池を使用します．

メモリマップ一覧表

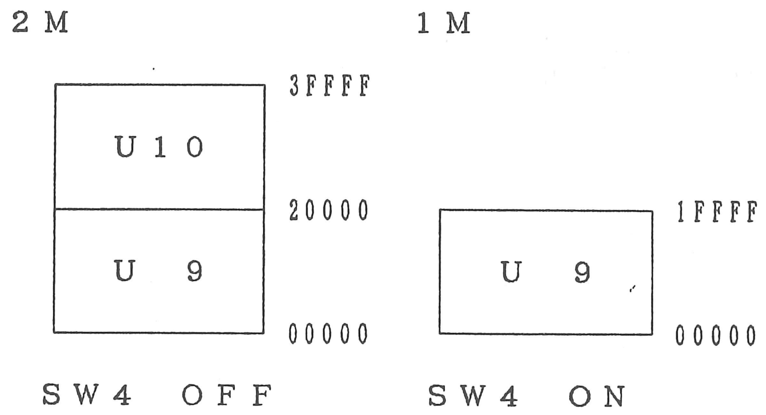
ＰＲＧメモリマップ

４Ｍ　　２Ｍ　　１Ｍ



４Ｍ	２Ｍ	１Ｍ
ＳＷ１　ＯＦＦ	ＳＷ１　ＯＦＦ	ＳＷ１　ＯＮ
ＳＷ２　ＯＦＦ	ＳＷ２　ＯＮ	ＳＷ２　ＯＮ

CHRメモリマップ



ＣＨＲメモリサイズが１Ｍ以下の場合は次ページ　ＰＲＧサイズ１Ｍ以下の場合の使用方法と同様の扱いとする